REMARKS

Claims 1-46 are pending in the present application. By this Response, claims 1, 6, 14, 15, 21, 30, 34 and 35 are amended, claims 2, 16-20, 23, and 36 are canceled and claims 47-51 are added. Specifically, claims 6, 14, 15, 30 and 35 are amended for clarification purposes only. Claims 1, 21 and 34 are amended to recite that the first/second profile information is processor profile information for the first/second SMP processor that identifies metrics associated with the execution of the first thread on the first/second SMP processor and is maintained by an operating system kernel. These claims are further amended to recite that the first thread profile information is maintained by a profiler application. No new matter has been added by the amendments to the claims or the additional claims 47-51. Support for these claims may be found in the specification, for example, at pages 63-76. Reconsideration of the claims is respectfully requested.

Amendments are made to the specification to update the cross-referenced application information. No new matter has been added by any of the amendments to the specification.

I. Telephone Interview

Applicants thank Examiner Vo for the courtesies extended to Applicants' representative during the June 14, 2004 telephone interview. During the telephone interview, Applicants' representative discussed the above amendments to the claims and the distinctions between the claims and the prior art. Examiner Vo indicated that the amendments to the independent claims appear to overcome the prior art of record but stated that a closer review of the references and application would be necessary before a final determination would be made. The substance of the interview is summarized in the following remarks.

II. <u>35 U.S.C. § 112, Second Paragraph</u>

The Office Action rejects claims 4, 6, 13-15 and 16-46 under 35 U.S.C. § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter, which applicants regard as the invention. This rejection is respectfully traversed. The claims are amended by this Response, where appropriate, to provide proper antecedent basis for the terms in the claims and to eliminate typographical errors. Therefore, Applicants respectfully request withdrawal of the rejection of claims 4, 6, 13-15 and 16-46 under 35 U.S.C. § 112, second paragraph.

III. <u>35 U.S.C. § 102, Alleged Anticipation</u>

The Office Action rejects claims 1, 2, 3, 5-6, 10, 13-16, 18, 20-23, 25-26, 30, 33-36, 38, 39, 43, and 46 under 35 U.S.C. § 102(b) as being allegedly anticipated by Summers (U.S. Patent No. 5,838,976). This rejection is respectfully traversed.

As to independent claim 1, the Office Action states:

> Regarding **claim 1**, Summers teaches a method for monitoring performance of a program being executed using symmetric multiprocessing (SMP) functionality (abstract) comprising:
>
> executing a native code routine (col. 3, lines 28-38);
>
> executing a first thread of the native code routine on a first symmetric multiprocessing (SMP) processor (col. 3, lines 28-54; the parent level, parent thread split into several child threads, col. 4. lines 10-24, col. 5, lines 17-48);
>
> ascertaining first profile information (4, lines 10-44, col. 5, lines 17-48);
>
> updating first thread profile information with the first profile information (col. 4, lines 10-44, col. 5, lines 17-48);
>
> executing the first thread of the native code routine on a second SMP processor (col. 4, lines 10-44, col. 5, lines 17-48);
>
> ascertaining second profile information (col. 4, lines 10-44, col. 5, lines 17-48); and
>
> updating first thread profile information with the second profile information (col. 4, lines 10-44, col. 5, lines 17-48).

Office Action dated March 15, 2004, page 5.

Claim 1 recites:

1. A method for monitoring performance of a program being executed using symmetric multiprocessing (SMP) functionality comprising:
executing a native code routine;
executing a first thread of the native code routine on a first symmetric multiprocessing (SMP) processor;
ascertaining first profile information, wherein the first profile information is processor profile information for the first SMP processor the identifies metrics associated with the execution of the first thread on the first SMP processor and is maintained by an operating system kernel;
updating first thread profile information, maintained by a profiler application, with the first profile information;
executing the first thread of the native code routine on a second SMP processor;
ascertaining second profile information, wherein the second profile information is processor profile information for the second SMP processor that identifies metrics associated with the execution of the first thread on the second SMP processor and is maintained by the operating system kernel; and
updating the first thread profile information, maintained by the profiler application, with the second profile information. (emphasis added)

A prior art reference anticipates the claimed invention under 35 U.S.C. § 102 only if every element of a claimed invention is identically shown in that single reference, arranged as they are in the claims. *In re Bond*, 910 F.2d 831, 832, 15 U.S.P.Q.2d 1566, 1567 (Fed. Cir. 1990). All limitations of the claimed invention must be considered when determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994). Anticipation focuses on whether a claim reads on the product or process a prior art reference discloses, not on what the reference broadly teaches. *Kalman v. Kimberly-Clark Corp.*, 713 F.2d 760, 218 U.S.P.Q. 781 (Fed. Cir. 1983). Summers does not identically show every element of the claimed invention arranged as they are in the claims. Specifically, with regard to claim 1, Summers does not teach ascertaining first profile information that is processor profile information for a first SMP processor that identifies metrics associated with the execution of the first thread on the first SMP processor and is maintained by an operating system kernel, updating first thread profile information, maintained by a profiler application, with the first profile information, ascertaining second profile information, wherein the second profile information is processor profile information for the second SMP processor that identifies metrics associated with the execution of the first thread on the second SMP processor and is maintained by an operating system kernel, or updating the first thread profile information, maintained by the profiler application, with the second profile information.

Summers teaches a system and method for profiling code on symmetric multiprocessor architectures. In the system of Summers, as illustrated in Figure 1 and described beginning at column 6, line 41 of Summers, memory storage areas 125-128 have at least four cells for each measured region in the process, e.g., four cells for each parent and childe thread. These cells are used to store metric values for the thread itself with regard to execution in direct and indirect parallelism and cumulative values of the metric for the thread alone and the thread with all of its children. The Summers system provides a parallel support layer (PSL) that permits the child threads to be associated with a parent thread in a manner such that the child thread knows which parent thread spawned it.

Summers is completely directed to using a profiler application to maintain metric values associated with the threads of a computer program being profiled. Even though Summers mentions that the threads may be executed on different processors in an SMP system (column 2, lines 10-13), Summers does not teach obtaining any metric information associated with individual processors within a SMP system and using this information to update the profile information maintained by a profiler application. Rather, Summers performs all metric gathering and storage with regard to threads. That is, all of the monitoring and collection of metric information is performed within a profiler and thus, is at a thread level. There is no ability in Summers to maintain, by an operating system, <u>processor</u> profile information at a processor level and then use this information to update <u>thread</u> level metric profile information. This is what is claimed in claim 1.

Claim 1 specifically states ascertaining first profile information that is <u>processor profile information</u> for a first SMP processor that <u>identifies metrics associated with the execution of the first thread on the first SMP processor</u> and is <u>maintained by an operating system kernel</u>, updating first thread profile information, <u>maintained by a profiler</u>

application, with the first profile information, ascertaining second profile information, wherein the second profile information is processor profile information for the second SMP processor that identifies metrics associated with the execution of the first thread on the second SMP processor and is maintained by an operating system kernel, or updating the first thread profile information, maintained by the profiler application, with the second profile information. In other words, a thread is executed on a first SMP processor and processor profile information, which is maintained by an operating system kernel, is ascertained which identifies metrics associate with the execution of the thread on the first SMP processor. This processor profile information is used to update thread profile information maintained by a profiler application. The same thread is also executed on a second SMP processor and processor profile information for the execution of this same thread on the second SMP processor is ascertained. This processor profile information, which is maintained by the operating system kernel, is then used to update the thread profile information maintained by the profiler application.

Summers, to the contrary, teaches that there are separate data areas in memory for each thread. A single thread data area is not updated based on metrics obtained regarding two different processors' execution of the same thread. This is because Summers performs profiling from a thread level with the profiling being completely performed by a profiler application. There is no ability in Summers to discern the metrics associated with the execution of a thread on a first processor from metrics associated with the execution of the same thread on a second processor. Moreover, there is no ability in Summers to update thread profile information based on processor profile information for two processors.

In addition to these distinctions with regard to independent claim 1, Summers also does not teach the specific features recited in dependent claims 5, 6, and 10. For example, with regard to claim 5, Summers does not teach retrieving processor accumulated profile information, retrieving processor last profile information, or calculating first profile information by comparing the processor accumulated profile information and the processor last profile information. The Office Action alleges that Summers teaches the features of claim 5 at column 4, lines 5-44 and in Figure 2. Column 4, lines 5-44 reads as follows:

> By using the present invention, a programmer or other observer can determine exactly how much CPU time was used by each thread or the sum of all threads executing at any level of a process. The best way to explain the operation of the present invention is to contrast it with the prior art.
>
> FIG. 2 shows a prior art code profiler that attempts to improve upon the naive scheme discussed in the Background section by accounting for different threads. FIG. 2 includes a code region 210, four threads 211-214, and a memory storage area 220 having a data cell 222. While only four threads 211-214 are shown in FIG. 2 and the other figures, it is important to recognize that a process can have any number of threads.
>
> Together, the four threads 211-214 symmetrically execute the region 210. Each thread has a corresponding CPU time delta 224-227. Each delta represents the amount of CPU clock time its corresponding thread 211-214 spent working on the code region 210. A thread typically determines its delta by sampling a CPU timer before and after the code region. The delta is the difference in time between the samples.
>
> The data cell 222 is simply a memory address within the memory storage area 220 capable of holding a delta value. Each thread 211-214 can access the value in cell 222. However, a thread must obtain exclusive access to cell 222 before the thread can change the value contained therein. To obtain exclusive access, a thread must lock cell 222. The locking process is called "synchronization." When a thread is finished updating cell 222, the thread releases the lock.
>
> In operation, the profiler of FIG. 2 directs each thread 211-214 to sample its CPU clock before and after the thread executes the code region. Then, each thread adds the time difference of the samples, or delta, to the value in cell 222. When all of the threads are finished, cell 222 contains the total CPU time used required to execute region 210.
>
> For example, assume that each thread in FIG. 2 has a delta of 10 CPU seconds. Upon completion, the total CPU time contained in cell 222 is 40 seconds. Thus, cell 222 contains the total amount of CPU time used by all of the threads in executing the measured code region.

This portion of Summers merely describes Figure 2 which teaches a prior art mechanism in which four threads are symmetrically executed and have a corresponding CPU time delta. The time deltas for each thread are accumulated to get a value that is stored in the data cell. Thus, the data cell stores the total amount of CPU time used by the threads.

There is nothing in this section, or any other section, of Summers that teaches to retrieve processor accumulated profile information, retrieve processor last profile information, and calculate profile information based on a comparison of the processor accumulated profile information and the processor last profile information. Even if the CPU time value stored in the data cell in Figure 2 of Summers were considered to be processor accumulated profile information, there is nothing in Summers that teaches processor last profile information or a comparison of the value in the data cell to processor last profile information to thereby calculate first profile information.

With regard to claim 6, Summers does not teach resetting processor last profile information by replacing the last processor profile information with the processor accumulated profile information. The Office Action alleges that this feature is taught by Summers at column 4, lines 17-44, which are reproduced above. Again, there is no teaching in this section, or any other section, of Summers with regard to processor last profile information or processor accumulated profile information. Moreover, there is no teaching in this section regarding resetting processor last profile information by replacing it with the processor accumulated profile information. All that is taught in the section cited by the Office Action is the adding of CPU time from a plurality of threads together and storing it in a data cell. Summers does not maintain any processor accumulated profile information or processor last profile information and thus, cannot perform the functions of claim 6 which make reference to this profile information.

Regarding claim 10, the Office Action fails to provide an explicit rejection of the features recited in claim 10 but instead merely states that claim 10 is rejected on the "same grounds as stated above" apparently making reference to the rejections of claims 1, 2, 5, 6 and 13. It is respectfully submit that claim 10 recites features that are not present in any of claims 1, 2, 5, 6 or 13 and the rejections of these claims in the Office Action fail to address these features of claim 10. Specifically, claim 10 recites receiving <u>virtual machine profile information</u> and updating <u>processor accumulated profile information</u> with <u>virtual machine profile information</u>. These features are not addressed by the Office Action with regard to any of claims 1, 2, 5, 6 or 13. Therefore, the Office Action has failed to establish a prima facie case of anticipation with regard to claim 10. Moreover, Summers does not, in actuality, teach receiving virtual machine profile information, updating processor accumulated profile information with virtual machine profile information, retrieving processor last profile information, or calculating the first profile information by comparing the processor accumulated profile information and the processor last profile information, as recited in claim 10.

With regard to independent claims 14, Summers does not teach retrieving first SMP processor accumulated profile information, retrieving first SMP processor last accumulated profile information, ascertaining first profile information from the first SMP processor accumulated profile information and the first SMP processor last accumulated profile information, updating first thread profile information with the first profile information, or setting first SMP processor last accumulated profile information in the first SMP processor data area equal to the first SMP processor accumulated profile information. As set forth above, Summers does not update thread profile information based on processor profile information. That is, as discussed above, Summers only performs profiling at a thread level and has separate metric storage areas for threads, not for processors in the SMP system and thus, cannot maintain processor profile information and cannot update thread profile information based on processor profile information.

Moreover, nowhere in Summers is there any teaching regarding processor accumulated profile information or processor last accumulated profile information and thus, there cannot be any teaching in Summers regarding the specific features of claim 14 which reference this profile information. As discussed above with regard to claim 5, the cited portion (column 4, lines 5-44) of Summers that allegedly teaches the features of an processor accumulated profile information and processor last accumulated profile information merely describes a process by which CPU time deltas for each thread are accumulated to get a value that is stored in a data cell. Even if the CPU time value stored in the data cell in Figure 2 of Summers were considered to be processor accumulated profile information, there is nothing in Summers that teaches processor last accumulated profile information or a comparison of the value in the data cell to processor last accumulated profile information to thereby ascertain first profile information.

Furthermore, there is no teaching in Summers to update thread profile information with profile information ascertained from comparing first processor accumulated profile information with first processor last accumulated profile information. In addition, there is nothing in Summers that teaches setting first SMP processor last accumulated profile information in a first SMP processor data area equal to the first SMP processor accumulated profile information. As discussed above, column 4 of Summers does not teach any of these features contrary to the allegations made by the Office Action. All that column 4 of Summers teaches is a prior art mechanism in which a single memory cell is used to maintain a total count of the processor time for a plurality of threads. There is no mention of any information that is even equivalent to first processor accumulated profile information, fist processor last accumulated profile information, or the specific features of claim 14 that operate on such information.

Similar distinctions apply to independent claim 15 which recites such features as "retrieving SMP processor accumulated profile information from each SMP processor's data area," "retrieving SMP processor last accumulated profile information from each processor's data area," "ascertaining profile information from the SMP processor accumulated profile information and the SMP processor last accumulated profile information for each thread running on a processor," "updating thread profile information with the profile information for each thread running on a processor," and "setting each SMP processor last accumulated profile information in the respective SMP processor's data area equal to the SMP processor accumulated profile information for that SMP processor." As stated above, Summers operates at a thread level and is not concerned with individual processor metrics. Thus, Summers does not teach processor accumulated profile information, processor last accumulated profile information, or updating thread profile information based on such processor profile information.

Moreover, Summers does not teach any processor data areas. Rather, the data storage cells described in Summers are specifically for individual threads, not processors. Summers has no mechanisms for storing metrics associated with processors, only threads. Therefore, there is no need in Summers for processor data areas in which processor accumulated profile information and processor last accumulated profile information would be stored.

Independent claims 21 and 34 recite similar features to that of claim 1 in their respective system and computer program product terminology. Therefore, claims 21 and 34 are distinguished over the Summers reference for similar reasons as set forth above with regard to claim 1. Similarly, claims 25, 26, 30, 38, 39 and 43 are distinguished over the Summers reference based on the same reasoning set forth above with regard to claims 5, 6 and 10.

In view of the above, Applicants respectfully submit that Summers does not teach each and every feature of claims 1-3, 5-6, 10, 13-16, 18, 20-23, 25-26, 30, 33-36, 38-39, 43, and 46 as is required under 35 U.S.C. § 102(b). Accordingly, Applicants respectfully request withdrawal of the rejection of claims 1-3, 5-6, 10, 13-16, 18, 20-23, 25-26, 30, 33-36, 38-39, 43, and 46 under 35 U.S.C. § 102(b).

IV. 35 U.S.C. § 103, Alleged Obviousness

The Office Action rejects claims 4, 7, 8, 9, 11, 12, 17, 19, 24, 27-29, 31, 32, 37, 40-42, 44, and 45 under 35 U.S.C. § 103(a) as being allegedly unpatentable over Summers in view of Liang (U.S. Patent Application Publication 2002/0099760 A1). This rejection is respectfully traversed for at least the same reasons as set forth above with regard to independent claims 1, 14, 15, 21 and 34. That is, Summers does not teach or suggest processor profile information, processor accumulated profile information, processor last accumulated profile information, or any of the specific steps/means/instructions that reference this information. Liang does not provide any teaching or suggestion to satisfy this deficiency in Summers.

Liang is directed to a method, apparatus and article of manufacture for time profiling multi-threaded programs. In the Liang system, a thread of execution of a multi-threaded program is selected and register data corresponding to the selected thread is retrieved. The register information is compared against stored register information for a previous suspension of the multi-threaded program and the selected thread is determined to be running if the computed register information is different from stored register information. Liang does not teach or suggest anything with regard to processor profile information that identifies metrics associated with the execution of the first thread on the first SMP processor and is maintained by an operating system kernel, as recited in claim 1. Liang does not teach or suggest updating thread profile information using processor profile information, as recited in claim 1. Liang does not teach anything with regard to processor accumulated profile information, processor last accumulated profile information, or determining first profile information based on a comparison of the processor accumulated profile information with the processor last accumulated profile information, as recited in claims 14 and 15. Moreover, Liang does not teach or suggest processor data areas in which processor accumulated profile information and processor last accumulated profile information are stored, as recited in claim 15.

Liang is cited by the Office Action as allegedly teaching, with regard to claims 7 and 8, processor accumulated profile information being implemented with a virtual machine at page 2, paragraph 0026 to page 3, paragraph 0028 and paragraphs 0036-0039. These sections of Liang merely teach that a multi-threaded program may be halted, a thread selected and corresponding register value retrieved to generate a checksum based on the register value. The register data is then compared to previous register information. If there is not a match, then the thread is running. The profiler that performs these operations may be part of a virtual machine (VM).

While Liang mentions that a VM may be used to perform profiling, there is nothing in Liang that teaches or suggests updating first processor accumulated profile information with virtual machine profile information, as recited in claims 7 and 8. Merely mentioning a virtual machine does not render this specific feature obvious. In fact, the virtual machine in Liang merely selects a thread, retrieves a register value and then compares the register value to a previous register value to determine if the thread is executing or not. There is no virtual machine profile information described in Liang nor is there any updating of processor accumulated profile information with virtual machine profile information in Liang.

Regarding claims 9, 29 and 42, the Office Action cites Liang as allegedly teaching updating processor accumulated profile information with virtual machine profile information and doing so with a virtual machine. Liang is also used to allegedly teach receiving a request to update processor accumulated profile information with virtual machine profile information and updating the processor accumulated profile information with the virtual machine profile information. The Office Action cites the same sections of Liang discussed above with regard to claims 7 and 8 as allegedly teaching these features. For similar reasons as set forth above, Liang does not teach these features and merely teaches comparing a register value for a selected thread to a previous register value to determine if the thread is executing. Other than merely mentioning the use of a virtual machine with a profiler, Liang has nothing to do with the specific features of any of the claims.

With regard to claims 11 and 12, the Office Action relies upon the alleged teachings of Summers with regard to the specific features in these claims. As discussed above, Summers does not teach or suggest processor data areas and thus, cannot be found to teach the specific features of claims 11 and 12.

With regard to claims 19, 24, 27, 28, 31, 32, 37, 40, 41, 44 and 45, the Office Action merely refers to its rejection of claims 7-9, 11-12, 17, 29 and 42 as the basis for the rejection of these claims. Thus, Applicants respond by stating that the rejection of these claims is traversed at least for the same reasons as noted above with regard to claims 7-9, 11-12, 29 and 42.

Thus, even if Liang were combinable with Summers, and one of ordinary skill in the art were somehow motivated to attempt such a combination, arguendo, the result still would not be the invention as recited in the pending claims. Therefore, Applicants respectfully submit that the rejection of claims 4, 7, 8, 9, 11, 12, 17, 19, 24, 27-29, 31, 32, 37, 40-42, 44, and 45 under 35 U.S.C. § 103(a) has been overcome. Accordingly, Applicants respectfully request withdrawal of the rejection of these claims.

V. <u>New Claims 47-51</u>

Claims 47-51 are added to recite additional features of the invention. For example, claims 47-50 are added to provide additional features with regard to the nature of the processor profile information, processor accumulated profile information and processor last accumulated profile information. Claim 51 recites a method of updating profiler information regarding threads of execution of a computer program based on processor metrics maintained in processor metric storage locations by an operating system kernel. These features are not taught or suggested by either of Summers or Liang.

VI. Conclusion

It is respectfully urged that the subject application is patentable over Summers and Liang and is now in condition for allowance. The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

Respectfully submitted,

DATE: June 15, 2004

[illegible]

Stephen J. Walder, Jr.
Reg. No. 41,534
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 367-2001
Attorney for Applicants